Quantum-Enhanced AI Self-Healing Network
Phase 1: Failure Detection & Prediction
Technical Documentation and Methodology Draft

Kishore Biswas: Technical Writer

January 20, 2026

1 Introduction

Phase 1: Failure Detection & Prediction, as outlined in the project roadmap. It includes:

- Documentation of the entire data pipeline.
- Detailed explanation of preprocessing and feature engineering.
- Specifications of Quantum Machine Learning (QML) algorithms with diagrams.
- Draft section for the thesis methodology chapter.

2 Documentation of the Entire Data Pipeline

The data pipeline is a hybrid quantum-classical system for network failure detection, starting from simulation and ending with predictions and alerts.

2.1 Pipeline Overview

The pipeline stages are:

1. **Network Simulation**: Use Mininet to create topology, generate traffic, and inject failures.
2. **Raw Data Capture**: Collect PCAP files with metrics like packet statistics, latency, loss, throughput, events.
3. **Preprocessing & Feature Engineering**: Clean, normalize, extract features, reduce dimensionality.
4. **Quantum Encoding**: Transform features into quantum states (e.g., angle encoding with RY gates).
5. **QML Model Execution**: Run models like QSVM or VQC for classification.
6. **Prediction & Decoding**: Interpret quantum measurements to classify failures with confidence scores.
7. **Monitoring & Alerts**: Visualize health, generate alerts, analyze history.
8. **Integration**: Connect via APIs to sandbox for real-time testing.

2.2 Key Implementation Details

- **Mininet Topology**: 4 switches (s1-s4), 8 hosts (h1-h8), partial mesh links with bandwidth 100 Mbps, delays 5-12ms. See Listing 1 for code.

Listing 1: Mininet Topology Code

```python
class CustomTopology(Topo):
    def __init__(self):
        Topo.__init__(self)
        # Add switches
        s1 = self.addSwitch('s1')
        s2 = self.addSwitch('s2')
        s3 = self.addSwitch('s3')
        s4 = self.addSwitch('s4')
        # Add 8 hosts with IP addresses
        for i in range(1, 9):
            host = self.addHost(f'h{i}', ip=f'10.0.0.{i}/24')
        # Connect switches in partial mesh
        self.addLink(s1, s2, bw=100, delay='5ms')
        self.addLink(s1, s3, bw=100, delay='10ms')
        self.addLink(s2, s4, bw=100, delay='7ms')
        self.addLink(s3, s4, bw=100, delay='12ms')
```

- **Failure Injection**: Table 1.

Table 1: Network Failure Injection Methods

Failure Type	Injection Method	Duration	Description
Link Failure	Link interface down	25 seconds	Complete disconnection between two switches
Packet Loss	TC qdisc netem loss	30 seconds	15% packet loss on switch interfaces
High Latency	TC qdisc netem delay	25 seconds	50ms additional latency on switch
Host Failure	Interface down	35 seconds	Complete host disconnection from network

- **Traffic Generation**: ICMP (ping), TCP (iPerf), UDP streams, HTTP requests.
- **Sandbox Integration**: Real-time monitoring, automated detection, model predictions, visualization.

3 Explanation of Preprocessing & Feature Engineering

Preprocessing transforms raw PCAP data into optimized features for QML input.

3.1 Steps

- **Data Cleaning**: Missing value imputation (mean), outlier removal, filtering, scaling (0-1).
- **Feature Extraction**: 14 features from PCAP (Table 2).
- **Dimensionality Reduction**: PCA, statistical analysis to reduce features.
- **Normalization**: StandardScaler for standardization.
- **Splitting**: 80/20 train-test with stratification.
- **Label Encoding**: 0: normal, 1: packet loss, 2: high latency, 3: link failure.

3.2 Feature Extraction Summary

Table 2: Feature Extraction Summary

Category	Feature Name	Description
Basic Statistics	packet count	Total packets in time window
	avg packet len	Average packet length in bytes
	std packet len	Standard deviation of packet lengths
Protocol Distribution	tcp ratio	Ratio of TCP packets
	udp ratio	Ratio of UDP packets
	icmp ratio	Ratio of ICMP packets
IP Characteristics	unique src ips	Number of unique source IPs
	unique dst ips	Number of unique destination IPs
Timing Features	avg inter arrival	Average time between packets
	inter arrival std	Std deviation of inter-arrival times
Entropy Features	packet len entropy	Shannon entropy of packet lengths
TCP Specific	tcp syn count	Ratio of TCP SYN packets
	tcp ack count	Ratio of TCP ACK packets
	avg tcp window	Average TCP window size

Data sources include flow statistics, performance metrics (latency, throughput), reliability (packet loss, error rate), congestion indicators, failure events.

4 QML Algorithm Specifications + Diagrams

QML models include QSVM, QBM, and VQC. QSVM is primary for Phase 1 due to stability and low qubit needs.

4.1 Model Selection Rationale

Table 3: QML Model Comparison

Model	Use Case	Advantages	Disadvantages
QSVM	Binary classification	Stable, hybrid-friendly, low qubit count	Limited to linear/polynomial kernels
QBM	Probabilistic modeling & prediction	Captures complex patterns, good for temporal	Complex training, noise sensitive, higher qubits

4.2 Quantum VQC Implementation

From Person 3:

Listing 2: Quantum VQC Circuit

```
# Create feature map and ansatz
feature_map = ZZFeatureMap(feature_dimension=n_qubits, reps=1)
ansatz = RealAmplitudes(num_qubits=n_qubits, reps=1)

# Create VQC
vqc = VQC(
    feature_map=feature_map,
    ansatz=ansatz,
    optimizer=COBYLA(maxiter=50),
    quantum_instance=Aer.get_backend('statevector_simulator')
)
```

Parameters: n$_qubits = 4, reps = 1, optimizer = COBYLA$.

4.3 Quantum Encoding

Transform classical features to quantum states using angle encoding: [f1, f2, f3] $\rightarrow$ [1, 2, 3] $\rightarrow$ RY() gates.

4.4 Hybrid Architecture Diagram

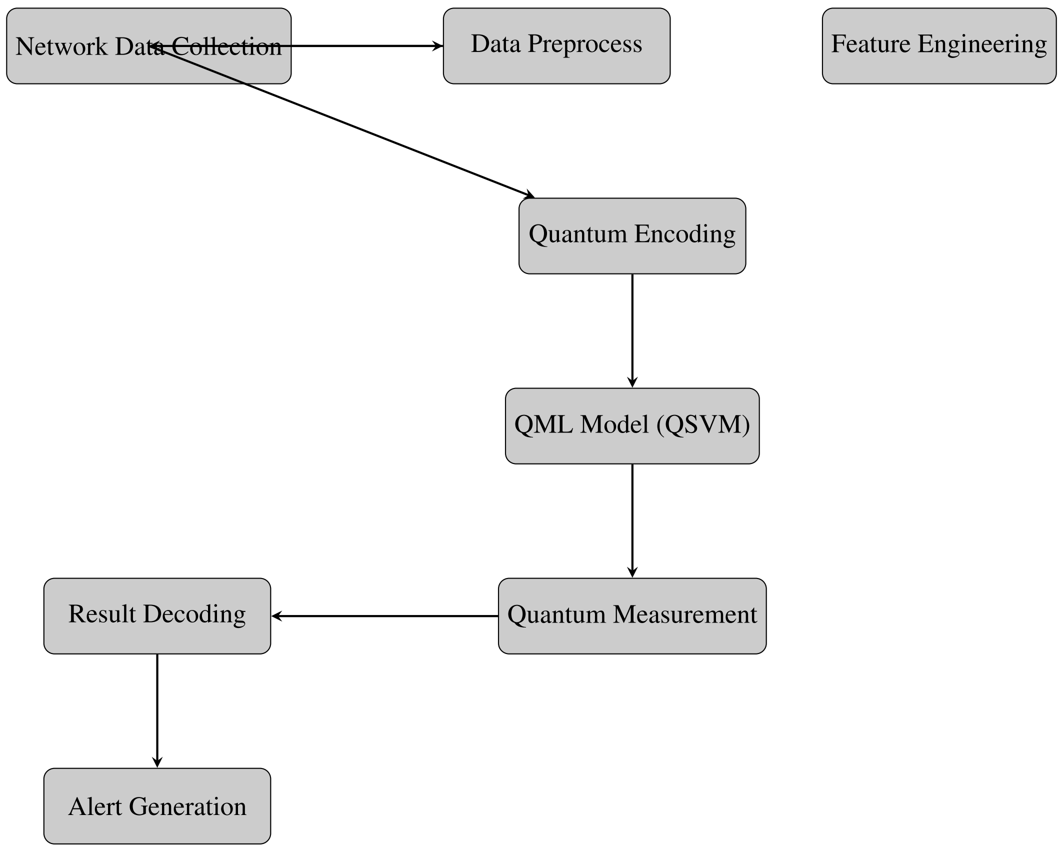

Figure 1: Hybrid Quantum-Classical Data Flow

4.5 Anomaly Classification

Table 4: Anomaly Classification Examples

Anomaly Type	Quantum Signature
DDoS Attack	$\rightarrow$, $\rightarrow$ /2 (High traffic, medium errors)
Link Failure	$\rightarrow$, $\rightarrow$ 0 (High latency, no traffic)
Hardware Fault	$\rightarrow$, random , (High errors)
Congestion	$\rightarrow$ 3/4, $\rightarrow$ /2 (High traffic & latency)

5 Draft Thesis Methodology Chapter Section

5.1 Chapter 3: Methodology - Phase 1: Failure Detection & Prediction

5.1.1 3.1 System Architecture Design

The system uses a hybrid quantum-classical architecture (Figure 1). Components: - Network Simulator (Mininet): Emulates environments for data generation. - Raw Network Data: Packet stats, latency, loss, throughput, events. - Preprocessing & Feature Engineering: Cleaning, normalization, extraction (Table 2), reduction. - QML Model: QSVM primary, with quantum encoding. - Prediction Output: Failure classification, confidence. - Monitoring Dashboard: Real-time visualization, alerts.

5.1.2 3.2 Data Pipeline

Simulation in Mininet with custom topology (Listing 1), failures (Table 1), traffic types. Data captured in PCAP, preprocessed, encoded, modeled, integrated in sandbox.

5.1.3 3.3 Preprocessing & Feature Engineering

Detailed in Section 3, with features in Table 2.

5.1.4 3.4 QML Models

QSVM for binary classification, VQC implemented (Listing 2). Comparison in model selection table.

5.1.5 3.5 Evaluation & Integration

Metrics: accuracy (100% for RF/SVM on synthetic), precision, recall, F1, ROC AUC. Limitations: synthetic data, computational resources. Future: real data, advanced circuits.